AMENDMENTS TO THE SPECIFICATION

IN THE SPECIFICATION:

Please amend paragraph [01] as follows:

-- [01] The present application is related to a co-pending U.S. application Ser. No. 10/051,558 ~~(Attorney Docket No. RSW020010101US1)~~, filed ~~concurrently herewith~~ on January 17, 2002, entitled "System and Method for Managing and Securing Meta Data", and assigned to the assignee of the present invention, which is herein fully incorporated by reference.--

Please amend paragraph [35] as follows:

--[35] In a preferred embodiment of the present invention, the central repository is accessed via known "Web-based Distributed Authoring and Versioning (WebDAV)" protocols, and supports the "ordered collections" and "locking" features of the WebDAV protocols known in the art. As known, WebDAV is an extension to the HTTP 1.1 protocol ~~(see, e.g., http://www.ics.uci.edu/pub/ietf/webdav/intro/webdav_intro.pdf)~~ and is implemented by a wide range of commercial repository products ~~(see, e.g., http://www.ietf.cnri.reston.va.us/rfc/rfc2518.txt for base protocol, http://www.ietf.cnri.reston.va.us/html.charters/webdav-charter.html for information on extensions and implementations)~~. In simple terms, WebDAV protocols allow a `client` to view a repository `server` as if it were an access controlled file system. A "userid" identifying a user (client) can be used to scope the files in the repository server which are available for manipulation as well as the operations that may be performed, and a "password" from a user can be used to authenticate the "userid" that a particular client claims. Based on the "userids" and "passwords", WebDAV protocols allow different users to access particular data from a central storage location (central repository) and to edit such data directly at that location. The "ordered collections" feature maintains modifications to the data at the central storage location in the collection order. To prevent different users from rendering modifications simultaneously, the WebDAV protocols provide the "locking" feature that allows only a single user to access a particular file at any given time. For instance, if user B desires to access a particular file when user A is currently accessing the file, the WebDAV system would block the access by the user B and inform user B of unavailability of the desired file. A more detailed discussion on WebDAV protocols and features can also be found at WebDAV's~~the~~ website~~ of http://www.webdav.org/~~. By using the existing WebDAV protocols in the central repository subsystem 60, the present invention eliminates the need to use special code typically used in prior art database systems.--

Please amend paragraph [36] as follows:

--[36] In still preferred embodiment, existing "RFC2069 Digest Access Authentication" protocols can be further implemented in the central repository subsystem 60 so that decryption keys and other access authorizing information would not be disclosed to network monitors. For instance, RFC2069 ~~(see, e.g., http://www.ietf.org/rfc/rfc2069.txt)~~ HTTP extension can be used in the process of authenticating the "userid" with the client's "password".--

Please amend paragraph [38] as follows:

--[38] The CDSA 30 is an existing security layer configuration for providing a widely-accepted set of layered security services defined by Intel Architecture Labs (IAL). Typically, the CDSA is implemented as computer software. Briefly, the functions and operations of the CDSA 30 will be discussed. The CDSA 30 includes a Common Security Services Manager (CSSM) API (application programming interface) that interacts with the applications 22-24 and the editor 25 to allow the applications 22-24 and the editor 25 to access the security services offered by the CDSA 30. The CDSA 30 also includes a plurality of service provider modules that offer these security services. Among the known service provider modules, the CDSA 30 may include a Cryptographic Service Provider (CSP) module, a Trust Policy (TP) module, a Certificate Library (CL) module, a Data storage Library (DL) module, and an Authorization Computation (AC) module, all known in the art. These modules provide services such as cryptographic operations including bulk encrypting and digital signature processing, accessing remote signing entities such as Certification Authorities (CA), storing certificates and cryptographic keys, etc. In addition, the CDSA 30, as known, includes elective module managers (EMM) that allow new services to be added easily. Under control of the EMM, new services can be added easily in a secure manner by merely providing new service provider modules as plug-ins that implement the new services. The process of adding and integrating the new service modules as plug-ins into the CDSA 30 is known in the art. More detailed operations and functions of the service provider modules and the CSSM API as well as the overall architecture of the CDSA 30 can be found at ~~the~~ Intel's website ~~of http://developer.intel.com/ial/security/~~.--

Please amend paragraph [38] as follows:

--[38] At the start of each user session at a computing device, the central database manager 17 requests the user to input "connection" information that will allow the central database manager 17 to connect via the communications network to the central repository and to input a "pass-phrase" (e.g., "BobsLongStringOfLettersAndNumbers") which is used to derive keys that will be used to decrypt/encrypt segments in the central repository and/or the local database(s) 15. In the preferred embodiment, the "connection" information needed to connect with the central repository includes: (1) the network name of the server holding the central repository ~~(e.g., "www.myrepository.com")~~, (2) a "userid" identifying the user (e.g., "bob"), and (3) a "password" associated with the user/userid (e.g., "letmein"). Two techniques can be used in the preferred embodiment to simplify such a user interaction. First, the "userid" and the network server name can be entered in an RFC822 style string that resembles an email address (e.g., bob@www.myrepository.com) and secondly, the "password" used to authenticate the user could be algorithmically derived from the "pass-phrase" already entered by the user using a secure one way hash or other cryptographic method.--

Please amend paragraph [53] as follows:

--[53] That is, in one embodiment, the encryption key is represented by a SHA1 hash of the new segment name/identifier, concatenated with the user's pass phrase or password. In another embodiment, the encryption key is represented by a SHA1 hash of the new segment name, concatenated with a SHA1 hash of the time portion of the new segment name, concatenated with the user's pass phrase or password. The segment name/identifier identifies the user session at the particular computing device, and can be represented as a string of some value or some other means. In one embodiment, the segment name/identifier can be a modified base 64 encoding of the time-date of the user session at which the first entry in the new segment is made. A SHA1 hash is generated using a SHA1 hash function well known in the cryptography field. A hash function is an existing technique of generating a "hash" based on an input value (e.g., the time portion of the new segment name). A hash represents a value of fixed length that is extracted from the input value using certain extraction rules. A SHA1 is one of different types of hash functions known in the art. A more detail discussion of a general hash function as well as a SHA1 hash function is provided at the website of the Centre for Applied Cryptographic Research (CACR) at the University of Waterloo.~~http://www.cacr.math.uwaterloo.ca/hac/~~ --